Installing the HCS12 Serial Monitor with additional permanent System Programs on Wytec MiniDragon+ boards

The compact serial monitor program *HCS12 Serial Monitor* (freescale, cf. application note AN2548/D) can be modified to allow installation of a number of small *system programs* in the protected area of Flash ROM. These programs can only be erased / overwritten through the Background Debug Mode (BDM) interface. They are therefore protected from being erased accidentally.

This small project uses switches SW1 and SW2 to choose between the serial monitor and either of two 'system programs' in the protected area of Flash: *SystemProgram1* and *SystemProgram2*. Both of these system programs can be implemented as regular C-functions (see the examples given in this project). Once compiled, their combined length must not exceed 6 kByte. Should this not be sufficient, the protected area can be increased from currently 8 kByte to the maximum value of 16 kByte. Upon reset, switch SW2 is checked to determine whether any of the system programs are to be run. If so (SW2 is pressed), execution is diverted to the *_Startup* code (source file Start12.c) which in turn calls *main* (source file main.c). The *main* program then checks the state of switch SW1 to determine which of the system programs is to be run: SystemProgram1 executes when SW1 is not pressed (the 7-segment display shows a '1'); SystemProgram2 is run when SW1 is pressed (the 7-segment display shows a '2').

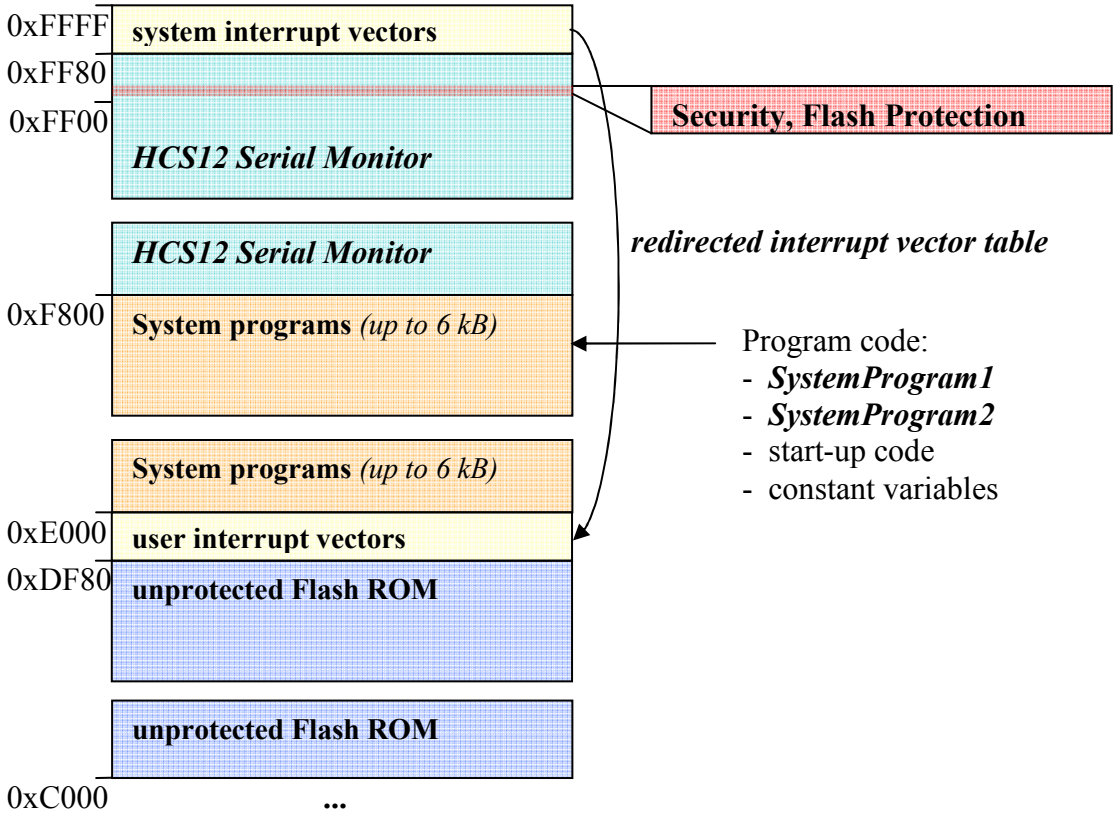

Figure 1 Memory map of the project

Figure 1 is an outline of the memory map of this project. The combined executable code of monitor and system programs has to fit into the area from 0xE000 to 0xFFFF (8kByte). The system vector table (0xFF80 – 0xFFFF) has been redirected to a *secondary vector table* located just below the bottom end of the protected ROM area (0xDF80 – 0xDFFF). During the download of a user program through the monitor program into unprotected Flash, the monitor ensures that the interrupt vectors specified by the user are placed into this secondary vector table. This is thus completely transparent to the user who can simply assume that the interrupt vectors reside in the area from 0xFF80 – 0xFFFF.

However, system programs are installed through the BDM interface which does not call upon the Flash programming routines of the monitor – after all, it is the monitor program itself we are trying to install right now. Therefore, any interrupt vector which may be used by a system program will have to be placed in the secondary interrupt table directly. This is done in source file isr_vectors.c, where the bases address of the vector table has been defined as 0xDF80. Note that neither of the two sample system programs of this project makes use of interrupts. With interrupt driven system program, source file isr_vectors.c would have to be added to the project.

In our laboratory we use the system programs for a short self-assessment routine which tests a number of digital I/O as well as analogue channels AD2 – AD15. We have designed a simple circuit board which protects the MC9S12DP256B/C from being damaged by over-voltages and short circuits. This board also includes 2 serially loaded D/A converters. The latter are used to test the analogue channels. The schematic of this circuit as well as the corresponding PCB layout masks are shown in Appendix A; a parts list is given in Appendix B. The self-assessment program displays an aggregate result of all tests on the on-board LCD display (pass/failure). A more detailed report can be obtained by connecting a serial terminal to SCI1 (115200 bps, 8 data bits, no parity, 1 stop bit, no protocol).

The second system program simply displays the board number on the LCD display. This allows us to sign out boards to individual students, making them accountable for any malicious damage that may occur...

Implementation

The protected Flash EEPROM area of the MC9S12DP256B (MiniDragon+) can only be programmed using a BDM interface. An inexpensive approach is to use two MiniDragon+ boards connected to each other via a 6-wire BDM cable. Alternatively, a MiniDragon+ and a Dragon-12 board can be used. The *master board* should run DBug-12 (installed by default) and be connected to the host via a serial communication interface (9600 bps, 8 bits, no parity, 1 stop bit, no flow control). A standard terminal program (e.g. HyperTerminal) can be used to control this board. The *slave board* is connected to the master board through a 6-wire BDM cable. Power only needs to be supplied to either the master or the slave board (i. e. one power supply is sufficient, see Figure 2).

The BDM interface of DBug-12 allows the target system (slave) to be programmed using the commands *fbulk* (erases the entire Flash EEPROM of the target system) and *fload* (downloads an S-Record into the Flash EEPROM of the target system). Unfortunately, DBug-12 requires all S-Records to be of the same length, which the S-Records produced by CodeWarrior are not. CodeWarrior has therefore been configured to run a small script file *(/bin/make_s19.bat)* which calls upon Gordon Doughman's S-Record conversion utility *SRecCvt.exe*. The final output file *(S12SerMon2r0.s19)* is the required S-Record in a downloadable format. You may have to re-run *make_s19.bat* if *S12SerMon2r0.s19* can not be found in the /bin folder after having build the project.

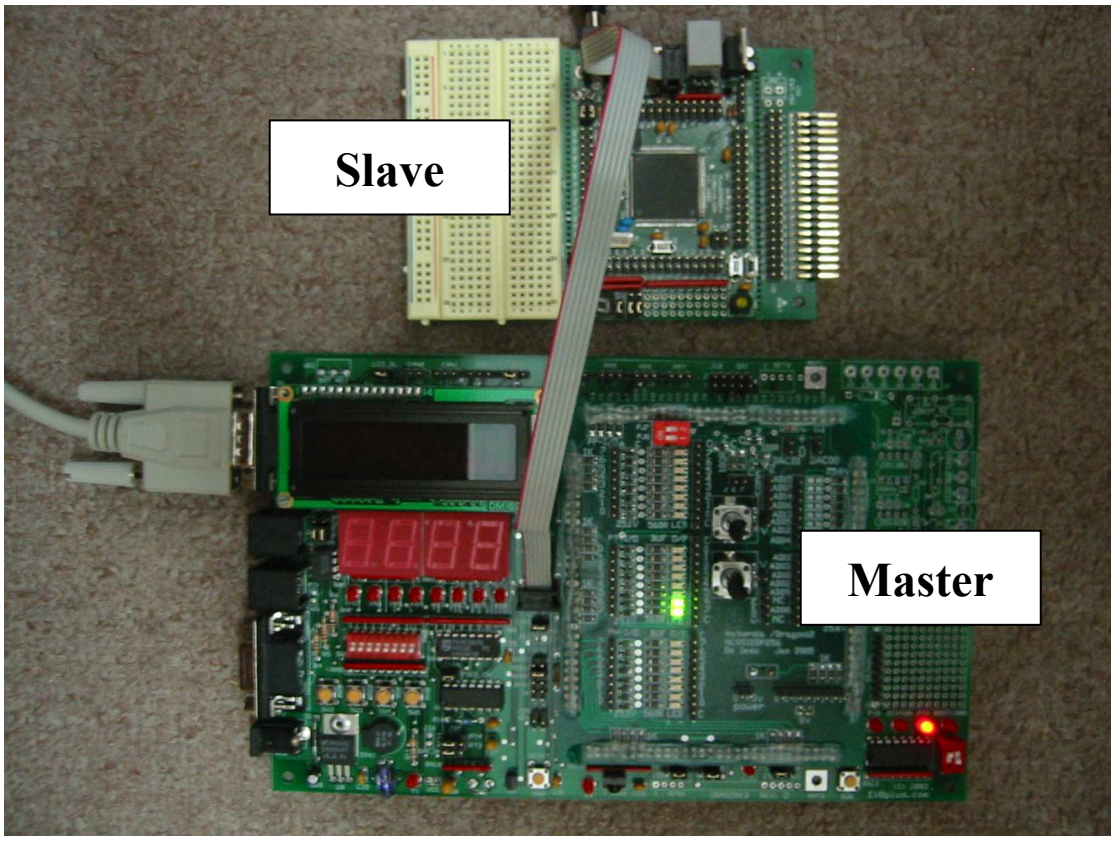

Figure 2 Using a Dragon-12 boards as BDM master module

Open the CodeWarrior project file and click on the green debug button of the CodeWarrior IDE to build the monitor program and convert the output file to a downloadable S-Record file.

Start HyperTerminal and reset the host board. Make sure the host board is set to *POD mode*. You should be presented with a small menu (Figure 3). Select menu item (2) to reset the target system. You should be presented with a prompt *'S>'* indicating that the target system is stopped.

Erase the Flash EEPROM of the target system by issuing the command *fbulk*. After a short delay, the prompt should reappear.

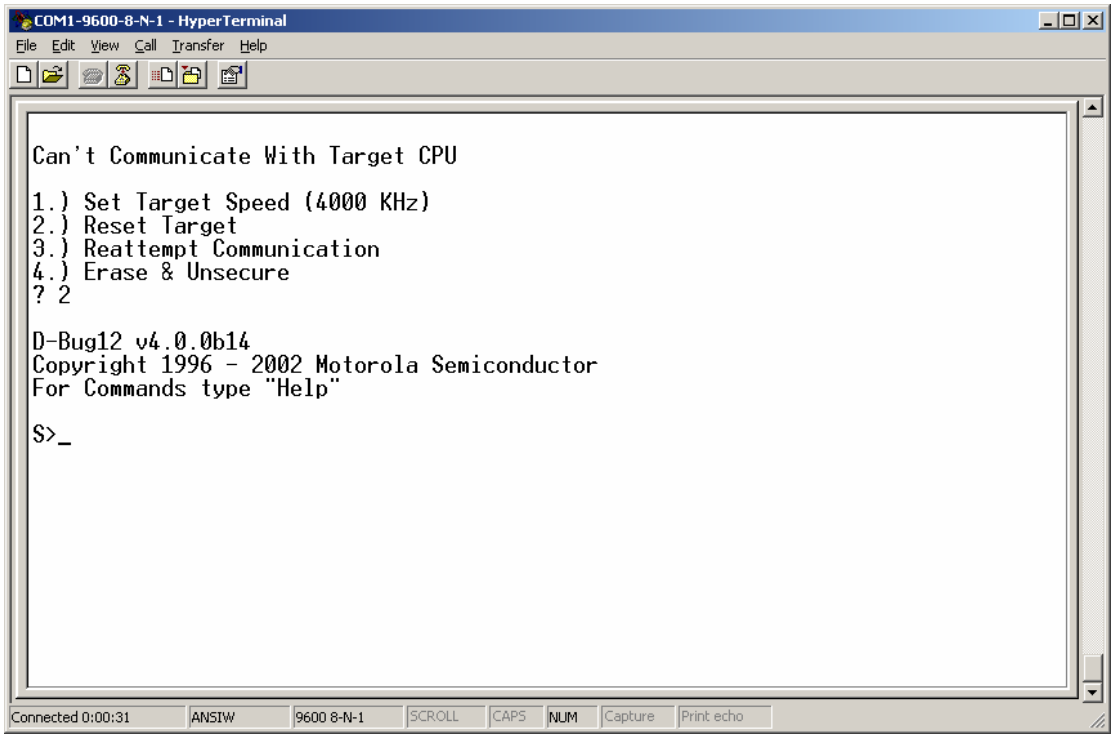

Figure 3 POD mode, host system

Issue the command *fload ;b* to download the S-Record file of the *HCS12 Serial Monitor*. Option *';b'* indicates that this file is in S1-format (as opposed to S2). From the *Transfer* menu select *Send Text File…* . Find and select the file *S12SerMon2r0.s19*. You will have to switch the displayed file type to *'All Files (*.*)'* (see Figure 4). Click on *Open* to start the download.

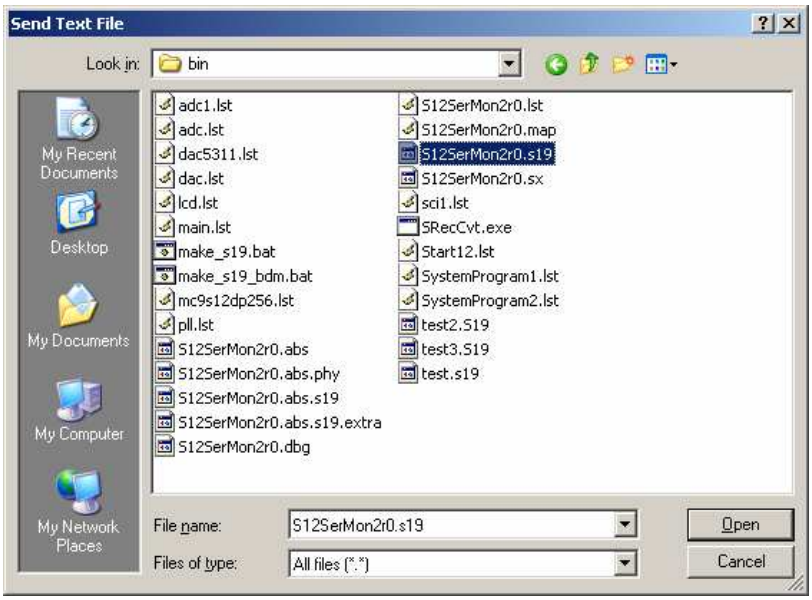

Figure 4 Downloading the *HCS12 Serial Monitor*

Once the download is complete, you should be presented with the prompt (Figure 5). The *HCS12 Serial Monitor* has successfully been written to the target board. Disconnect the BDM cable from the target system and close the terminal window. The board is now ready to be used. Note: The total number of S-records downloaded to the target depends on the size of your system programs. Figure 5 shows the case of no system programs.

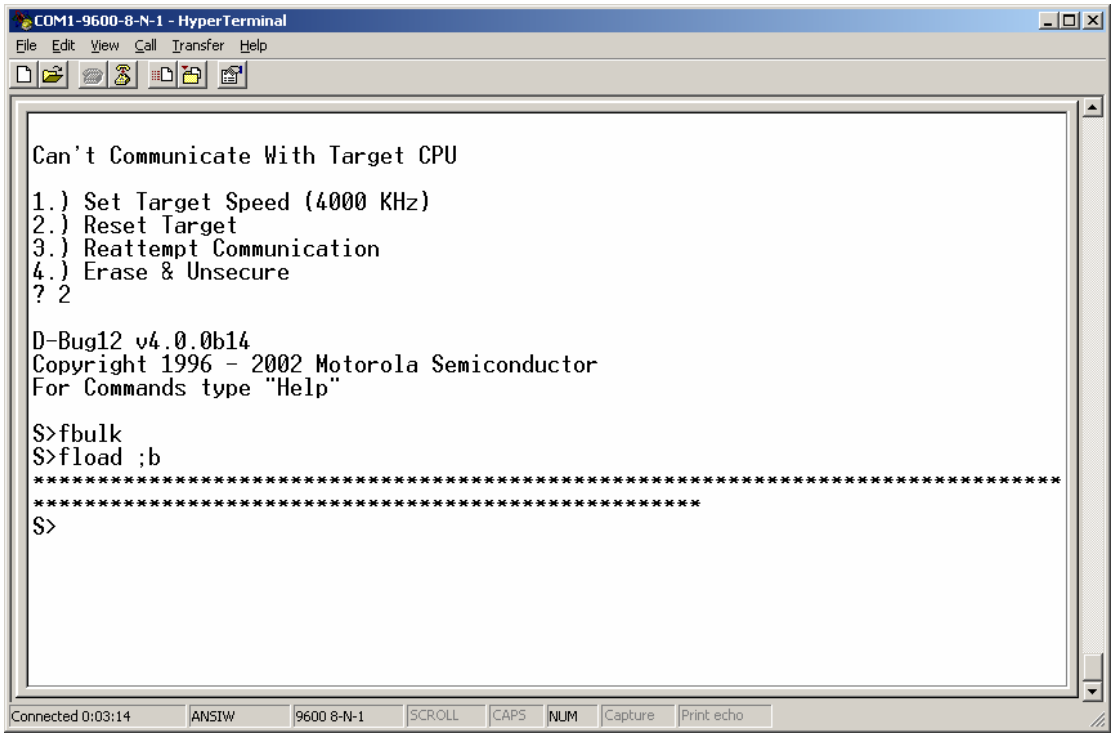

Figure 5 Download complete

An alternative to the download of the monitor program via a second MiniDragon+/Dragon12 board is to use P&E Microsystems's *BDM-Multilink* cable and the corresponding Flash programming software *PROG12Z*.

The advantage of using the Background Debug Mode (BDM) interface of the microcontroller is a much reduced programming time. For microcontrollers with paged memory addressing (e.g. the MC9S12DP256B/C), PROG12Z expects that all S-Records use a linear physical memory space (as opposed to the *logical addresses* with page number and page offset address). This is where *PROG12Z* differs from *D-Bug12*. We therefore have to convert the original S-Record produced by the project (/bin/S12SerMon2r0.abs.s19) to a slightly different format as what is required when using a second Dragon12 board running *D-Bug12*. The main difference is that *PROG12Z* requires Flash EEPROM addresses to be offset by 0xF0000 (S2 records), whereas *D-Bug12* uses unmodified S1 records to directly represent physical addresses in segment 0. As before, we will make use of Gordon Doughman's S-Record conversion utility *SRecCvt.exe*. For consistency with other conversion utilities (e.g.

P&E Microsystems's PPAGE Logical to Physical S-Record Conversion Program *log2phy_12)* we will request records with a length of 16 bytes.

The following command line achieves this:

```
SRecCvt -s2 -m 0 fffff 16 -of F0000 -o S12SerMon2r0.bdm.s19 S12SerMon2r0.abs.s19
```

This requests S2 records with a length of 16 bytes and offset by 0xF0000; the input file is S12SerMon2r0.abs.s19 and the output file is S12SerMon2r0.bdm.s19. The latter can now be loaded into PROG12Z and written to the Flash EEPROM of the MC9S12DP256B/C. Note that project folder /bin contains a small batch file *(make_s19_bdm.bat)* which performs the above conversion. The project has been set up to run this file automatically. Should this not be the case (e.g. you may have changed the configuration), please ensure to run this batch file manually. Double-clicking its icon in Windows Explorer runs this script and produces the required S-Record file.

Connect the BDM-Multilink Cable to the MiniDragon+ connector labelled *BDM in*. The power LED on the BDM-Multilink should be on – if not, the polarity of the 6-wire BDM cable may be incorrect and should be swapped (Figure 6).

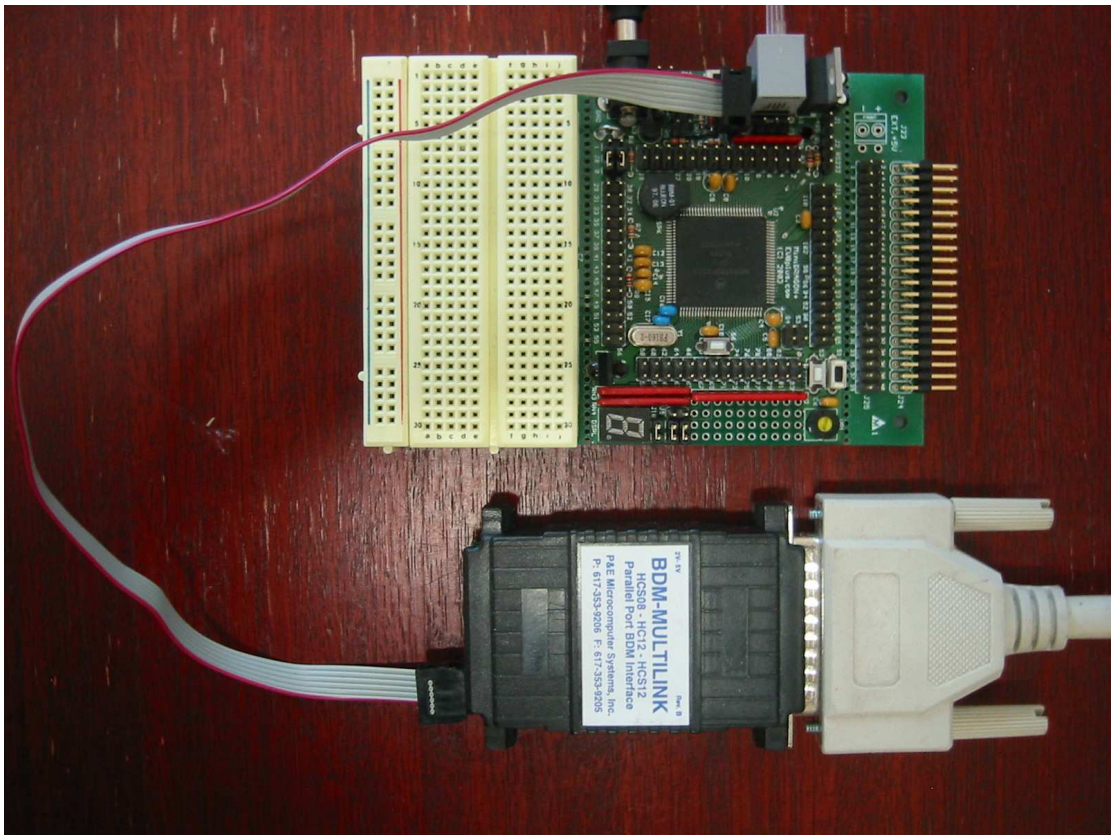

Figure 6 Connecting the MiniDragon+ to a P&E BDM-Multilink cable

Figure 7 shows the user interface of PROG12Z. Press the push button with the small yellow bulb (*Reset Processor*, second from the left). This resets both the software interface of PROG12Z as well as the BDM interface of the microcontroller.

Now, click on the next push button (Choose Module, symbolic representation of a microchip). A file requester appears (Figure 8); choose the programming algorithm definition file *9S12dp256_256k.12P*.

Erase the module by clicking on the push button with a symbolic representation of a yellow pencil / red eraser. This unlocks and erases the entire Flash EEPROM of the MiniDragon+. The microcontroller is now ready to be programmed with the monitor program / system programs.

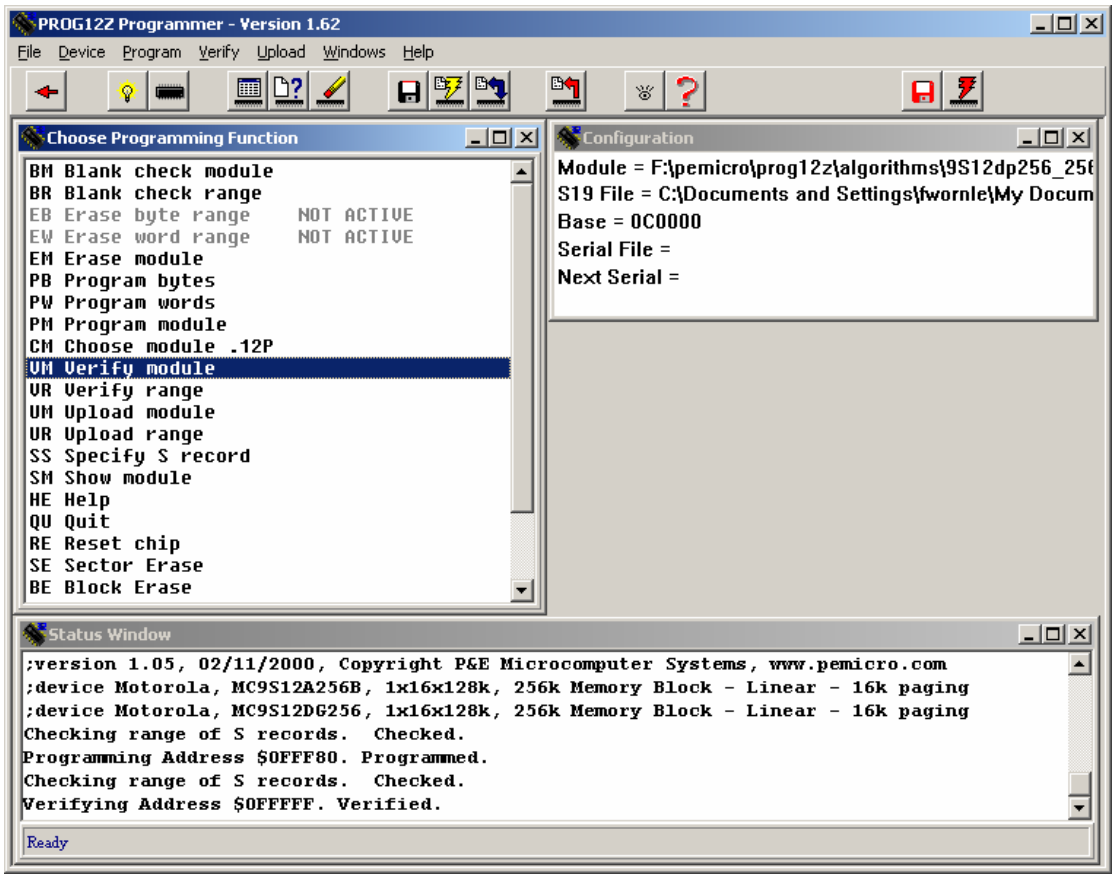

Figure 7 Programming the Flash with P&E Microsystems's PROG12Z

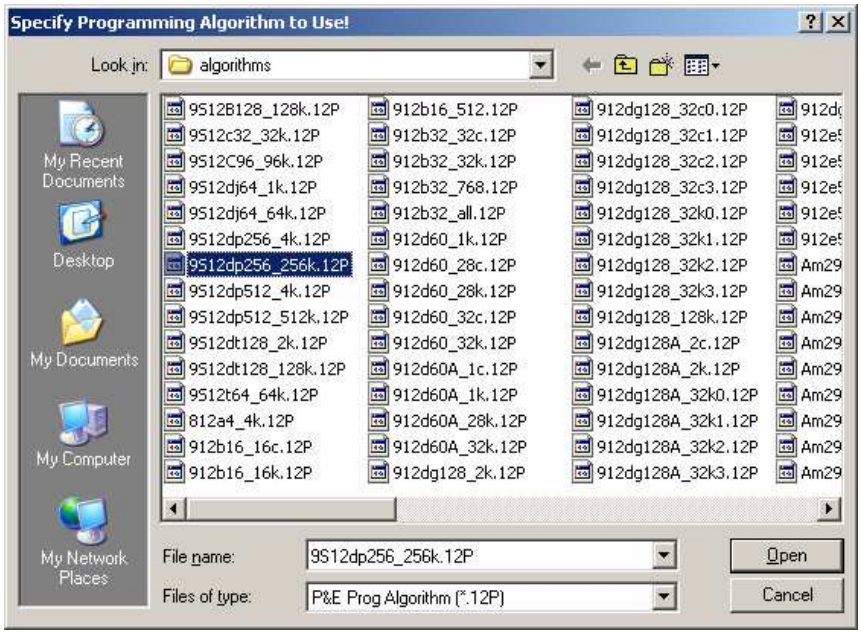

Figure 8 Choosing the programming algorithm

Click onto the small black floppy-disc symbol to open another file requester; open the /bin folder of the project and select the newly created S-Record *S12SerMon2r0.bdm.s19* (Figure 9).

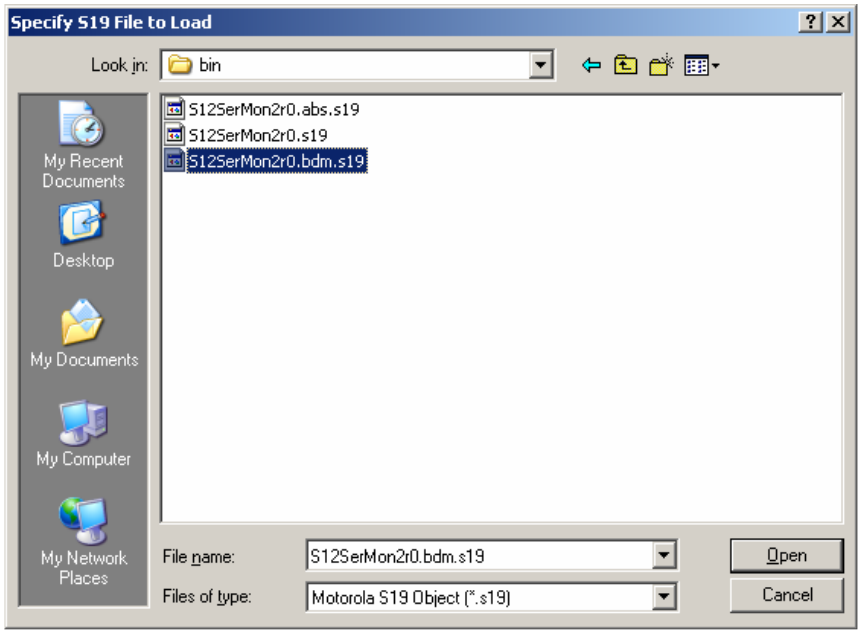

Figure 9 Selecting the S-Record file to be programmed

Program the microcontroller by clicking onto the programming push button (small yellow flash symbol, next to the floppy-disc symbol). The status window of PROG12Z should count up the addresses which have been programmed. Click onto

the *verify* button (blue arrow, next to the programming button). The status window should display the programmed address range and indicate the successful programming of the chip (Figure 10). The entire process should only have taken 1 – 2 seconds.

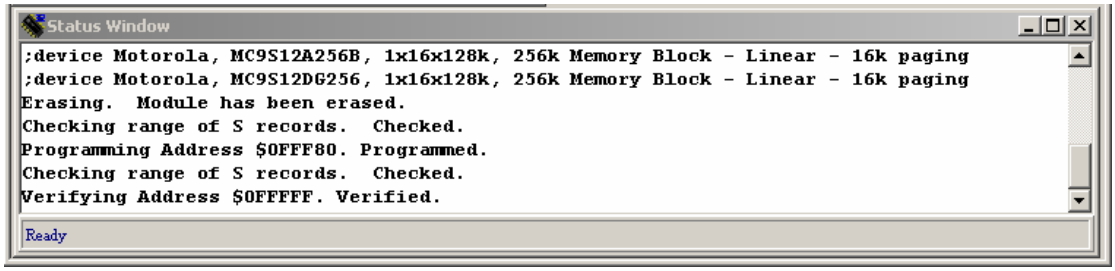

Figure 10 The status window of PROG12Z